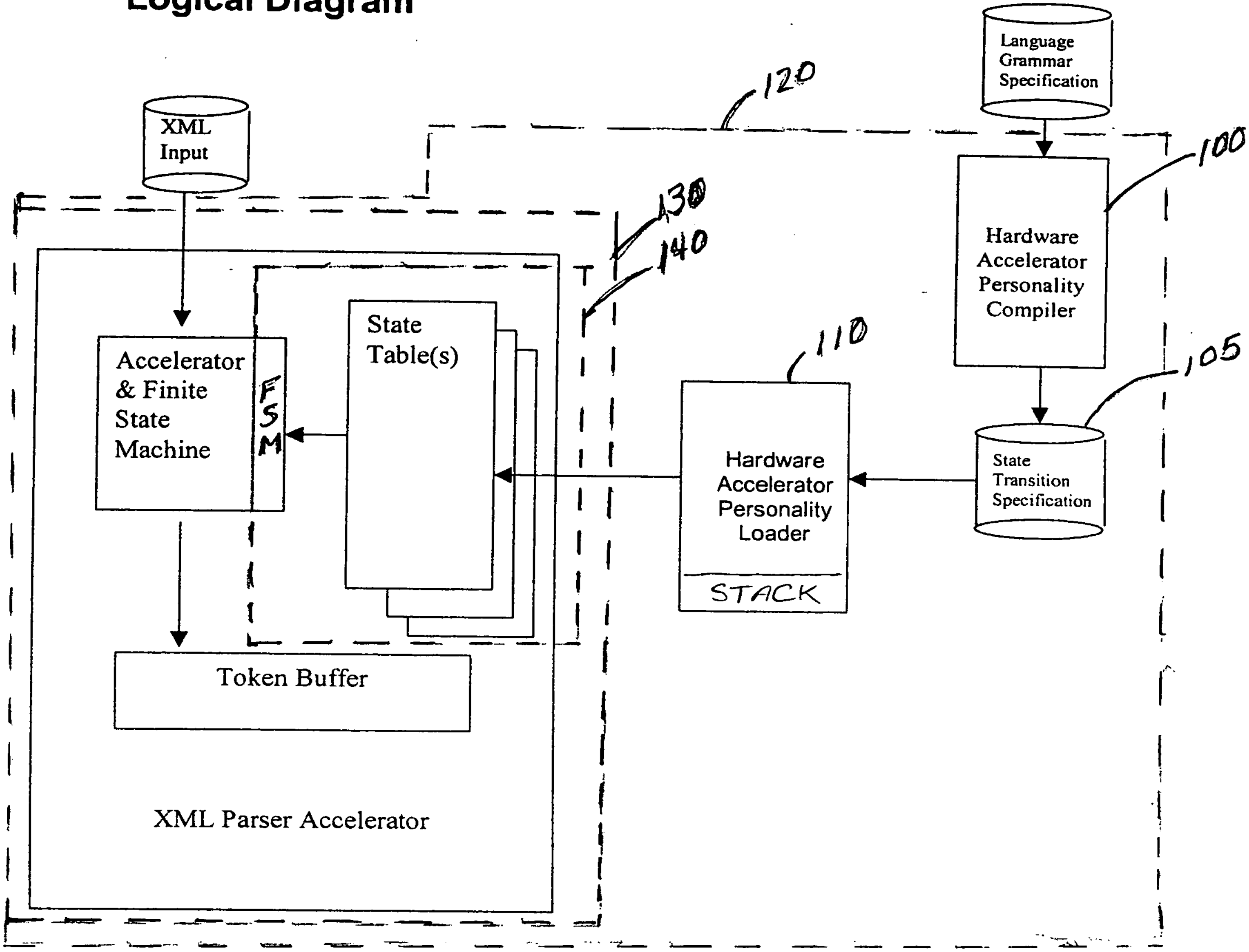
Logical Diagram



Language Grammar Specification
120
100
XML Input
130
140
Hardware Accelerator Personality Compiler
State Table(s)
110
105
Accelerator & Finite State Machine
FSM
Hardware Accelerator Personality Loader
State Transition Specification
STACK
Token Buffer
XML Parser Accelerator

FIGURE 1

XML Parser State Table Example

ASCII Value

Index → Base Address ↓	00	01	02	03	………..	FD	FE	FF
State 0	Stay in state 0	Stay in state 0	**Go To state 1**	Stay in state 0	………..	Stay in state 0, SPECIAL INTERRUPT	Stay in state 0	Stay in state 0
State 1	Stay in State 1	**Go To State 2**	Go To state 0	Stay in state 1	………..	**Go To state 4**	Go To State 0	Go To State 0
State 2	Stay in State 2	Go To State 3	Go To State 0	**Go To State 3, Special Interrupt**	………..	Go To State 0	Go To State 0	Go To State 0
State 3	Stay in State 3	Go To State 0	**Go To State 0, Store Token**	Go To State 0	………..	Go To State 0	Go To State 0	Go To State 0
State 4	Stay in State 4	**Go To State 0, Store Token**	Go To State 0	Go To State 0	………..	Go To State 0	**Go To State 5**	Go To State 0
State 5	Stay in State 5	Go To State 0	Go To State 0	Go To State 0	………..	Go To State 0	Go To State 0	**Go To State 0, Store Token**

FIGURE 2A

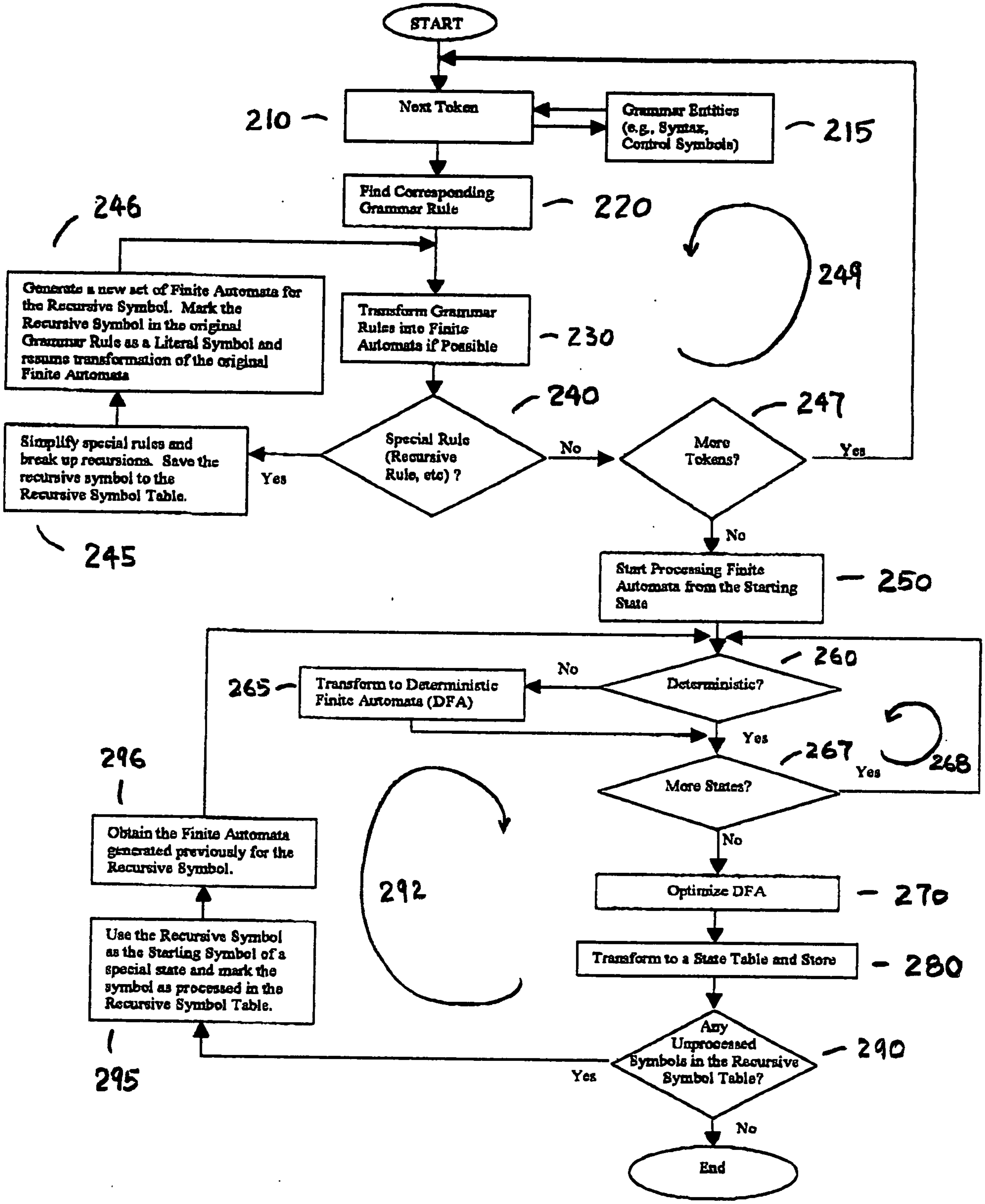

START
Next Token
210
Grammar Entities (e.g., Syntax, Control Symbols)
215
Find Corresponding Grammar Rule
220
246
Generate a new set of Finite Automata for the Recursive Symbol. Mark the Recursive Symbol in the original Grammar Rule as a Literal Symbol and resume transformation of the original Finite Automata
Transform Grammar Rules into Finite Automata if Possible
230
249
240
247
Simplify special rules and break up recursions. Save the recursive symbol to the Recursive Symbol Table.
Yes
Special Rule (Recursive Rule, etc) ?
No
More Tokens?
Yes
245
No
Start Processing Finite Automata from the Starting State
250
260
265
Transform to Deterministic Finite Automata (DFA)
No
Deterministic?
Yes
267
268
Yes
296
More States?
No
Obtain the Finite Automata generated previously for the Recursive Symbol.
292
Optimize DFA
270
Use the Recursive Symbol as the Starting Symbol of a special state and mark the symbol as processed in the Recursive Symbol Table.
Transform to a State Table and Store
280
Any Unprocessed Symbols in the Recursive Symbol Table?
290
295
Yes
No
End

Figure 2B

High Level Logic Flow Diagram

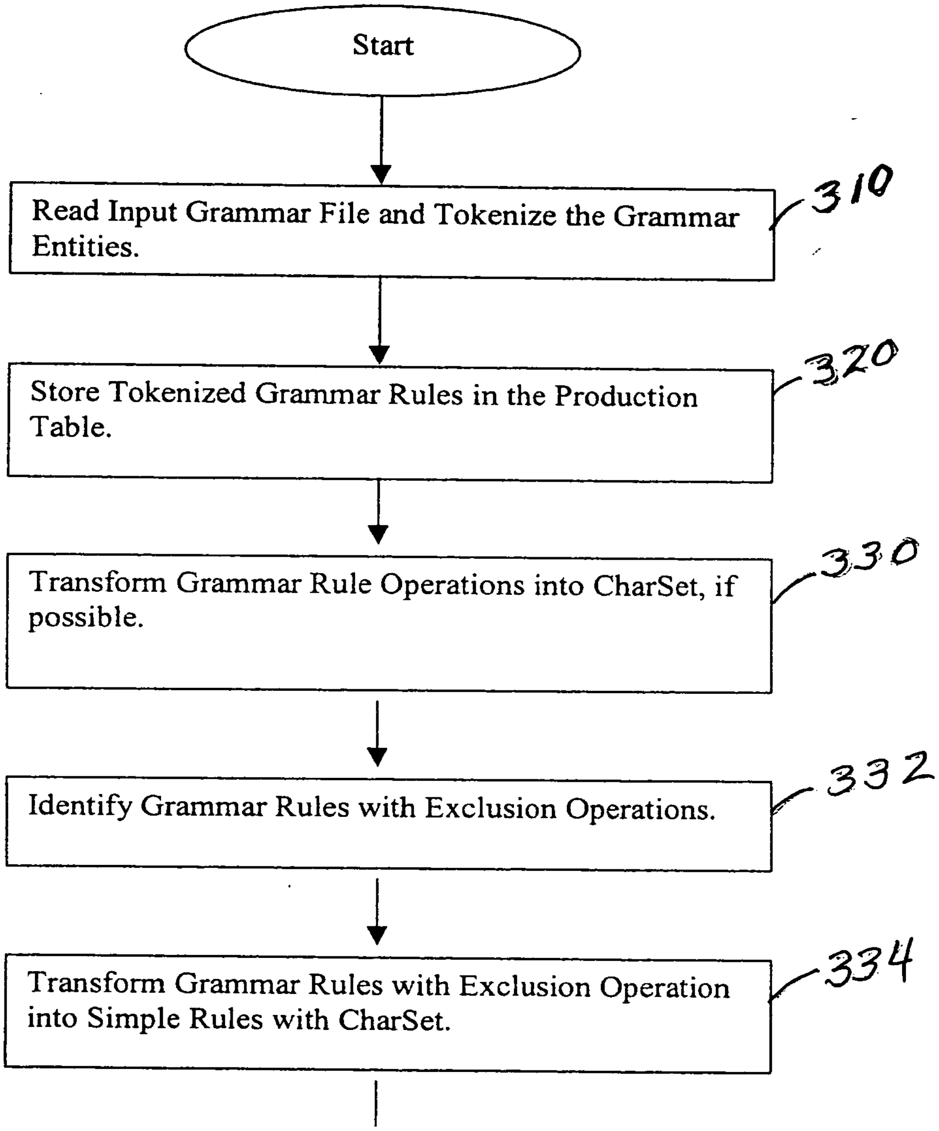

Start
Read Input Grammar File and Tokenize the Grammar Entities.
310
Store Tokenized Grammar Rules in the Production Table.
320
Transform Grammar Rule Operations into CharSet, if possible.
330
Identify Grammar Rules with Exclusion Operations.
332
Transform Grammar Rules with Exclusion Operation into Simple Rules with CharSet.
334

Figure 3: High Level Logic Flow Diagram

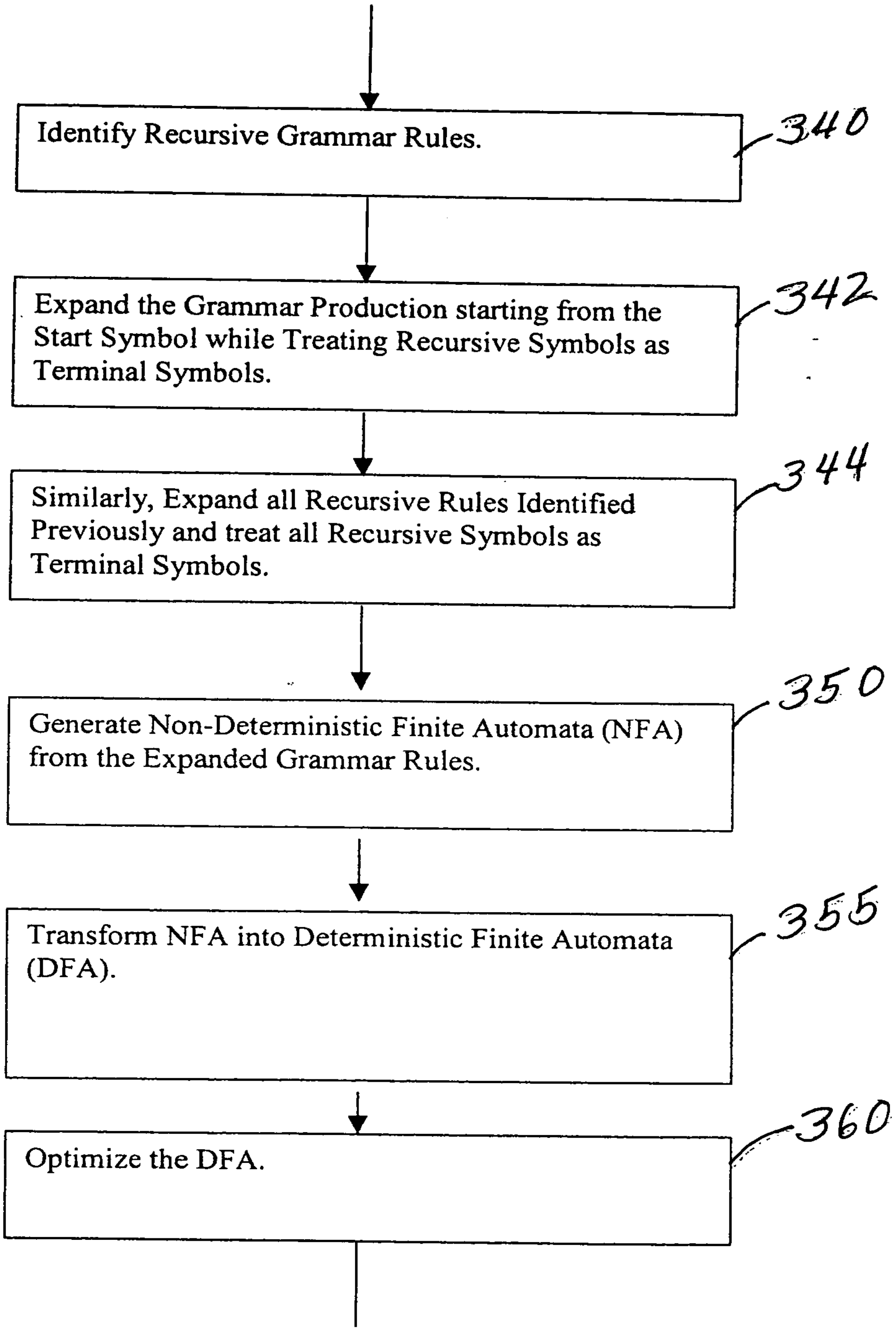

Identify Recursive Grammar Rules.
340
Expand the Grammar Production starting from the Start Symbol while Treating Recursive Symbols as Terminal Symbols.
342
Similarly, Expand all Recursive Rules Identified Previously and treat all Recursive Symbols as Terminal Symbols.
344
Generate Non-Deterministic Finite Automata (NFA) from the Expanded Grammar Rules.
350
Transform NFA into Deterministic Finite Automata (DFA).
355
Optimize the DFA.
360

Figure 3: High Level Logic Flow Diagram (Continued)

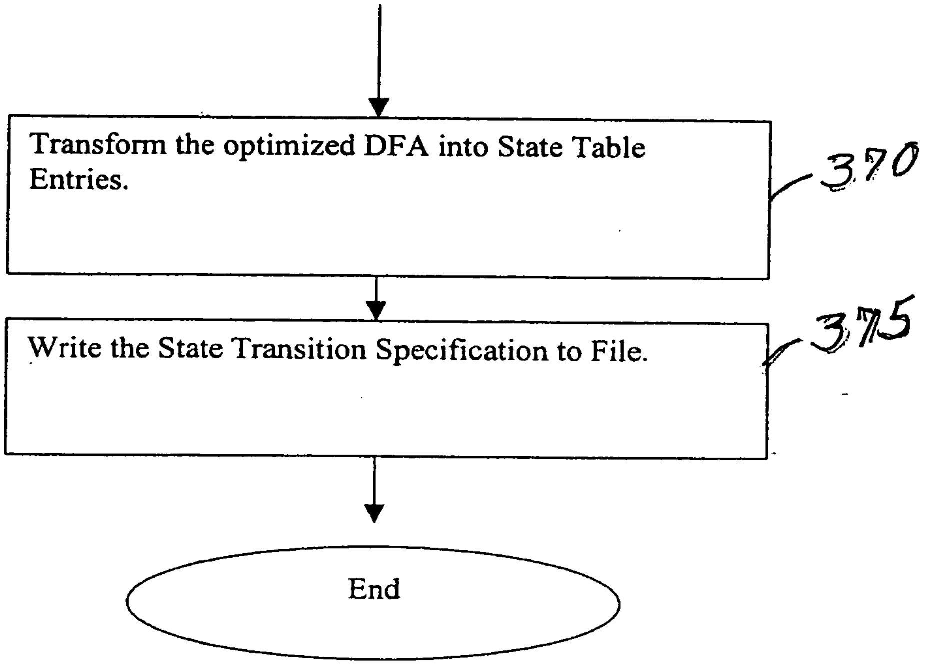

Transform the optimized DFA into State Table Entries.
370
Write the State Transition Specification to File.
375
End

Figure 3: High Level Logic Flow Diagram (Continued)

Object Context Diagram for Hardware Accelerator Personality Compiler (HAPC)

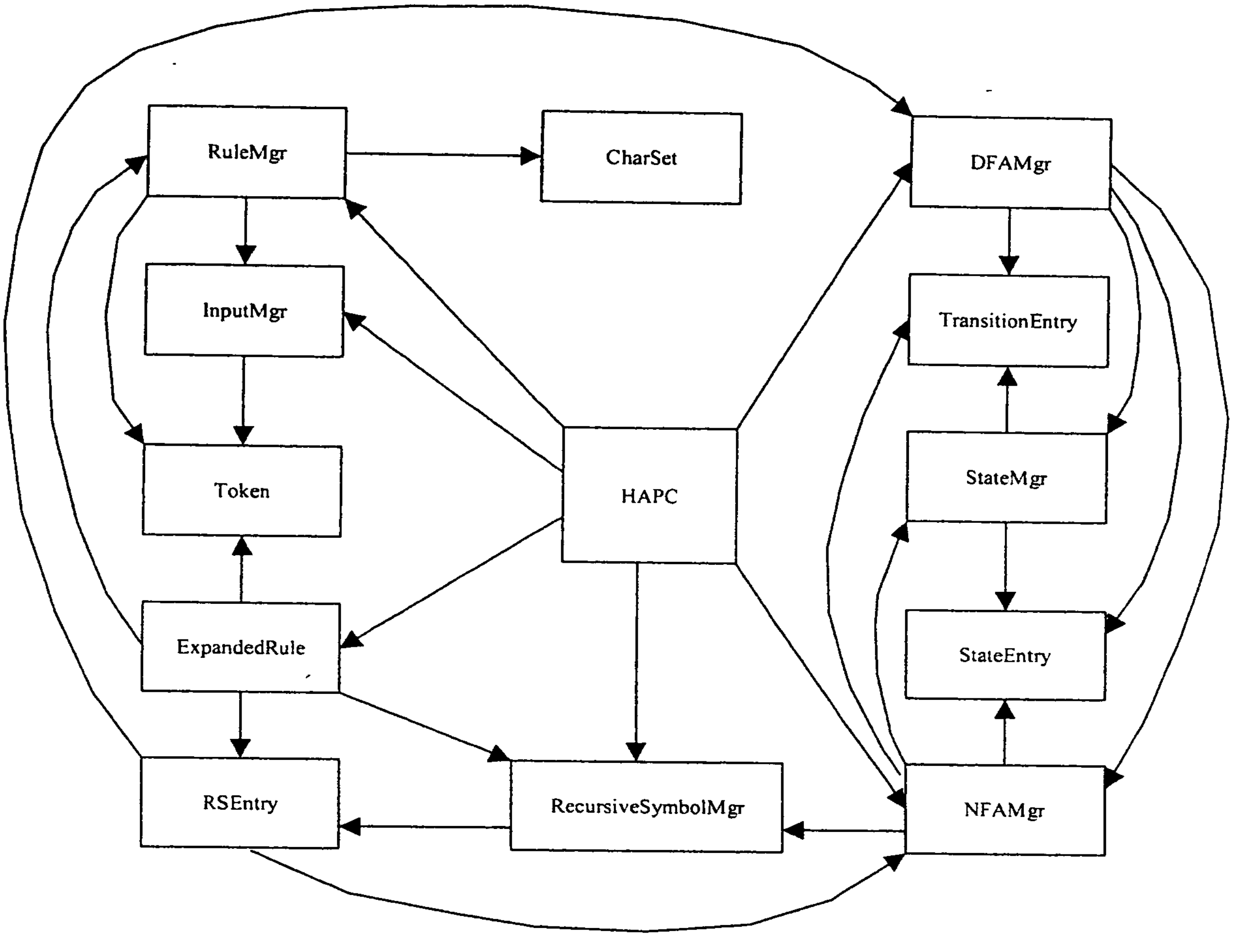


Figure 4: High Level Context Diagram

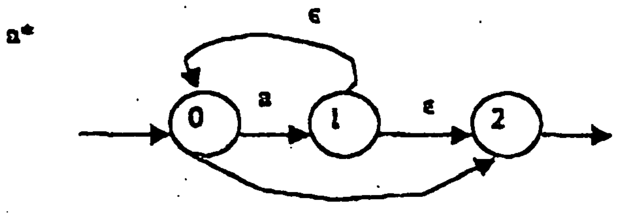


FIG 5A – Zero or More Occurances of a

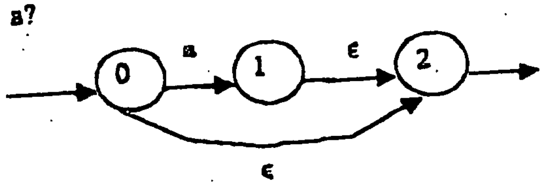


FIG 5B – Zero or One Occurance of a

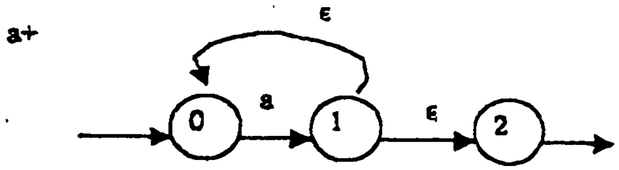


FIG 5C – One or More Occurances of a

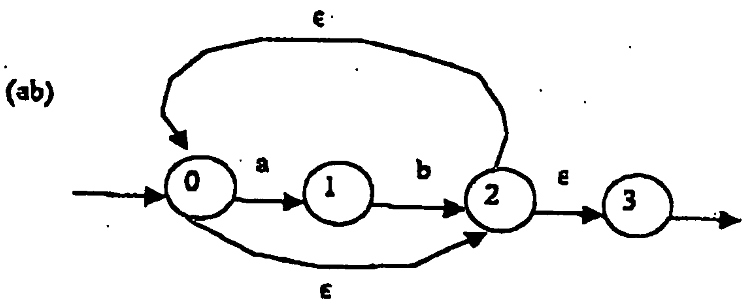


Fig 5D - Zero or More Occurances of the sequence a followed by b

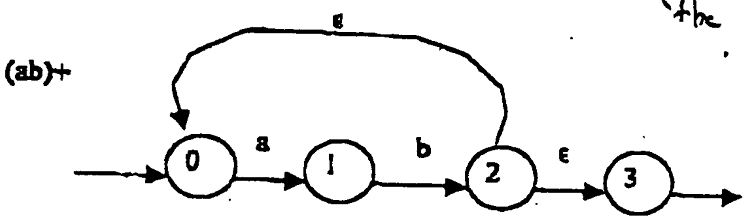


Fig 5E - One or More Occurances of the Sequence a followed by b

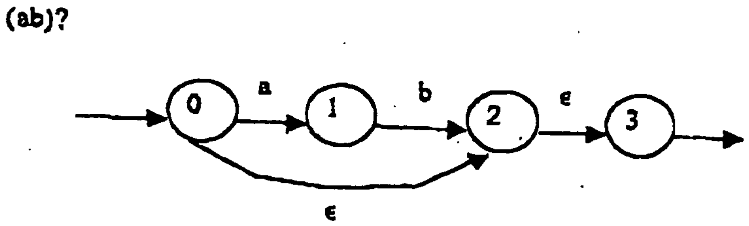


Fig 5F - Zero or One Occurance of the Sequence a followed by b

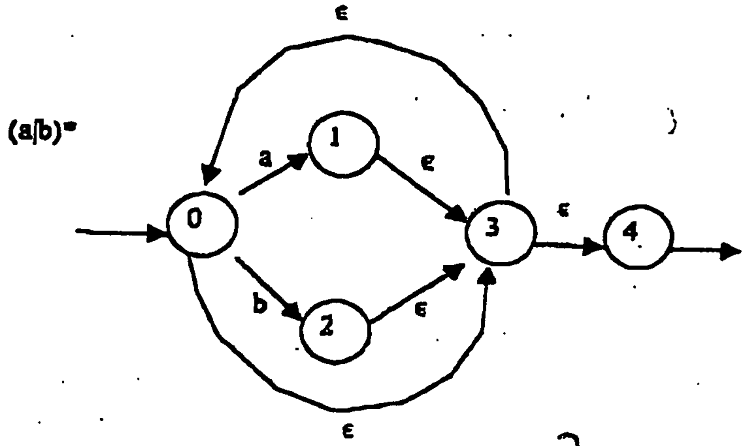


Fig. 5G - Zero or More Occurances of either a or b but not both

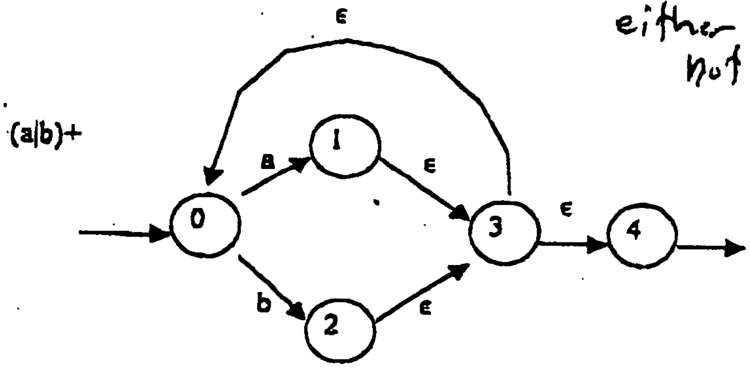


Fig 5H - One or More Occurances of either a or b but not both

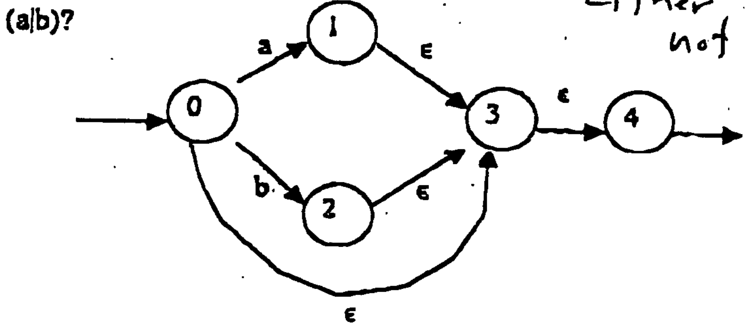


Fig. 5I - Zero or One Occurance of either a or b but not both

```
                  <?xml version="1.0"?>
                  <!DOCTYPE HAPC SYSTEM "hapc.dtd">
                  <HAPC>
                   <StateTransitionSpecification>
600                 <Timestamp>2002-10-30 17:17:53.0</Timestamp>
                    <GrammarFilename>xml_bnf_override.hapc</GrammarFilename>
                    <StateTableGeneralInfo>
        611      -   <TableCount>5</TableCount>
                     <TableNames>
                      <MainTableName>Main</MainTableName>
        612           <RecursiveSymbolTableName>seq</RecursiveSymbolTableName>
                      <RecursiveSymbolTableName>choice</RecursiveSymbolTableName>
                      <RecursiveSymbolTableName>element</RecursiveSymbolTableName>
                      <RecursiveSymbolTableName>cp</RecursiveSymbolTableName>
                     </TableNames>
610                  <StateTableColumnInfo>
                      <ColumnCount>128</ColumnCount>
                      <ColumnName>
                       <DecimalCode>0</DecimalCode>
                      </ColumnName>
                        .
        613             .
                        .
                      <ColumnName>
                       <DecimalCode>32</DecimalCode>
                       <PrintableName> </PrintableName>
                      </ColumnName>
                        .
                        .
                        .
                      <ColumnName>
                       <DecimalCode>127</DecimalCode>
                      </ColumnName>
                     </StateTableColumnInfo>
                    </StateTableGeneralInfo>
                    <StateTableDetailedInfo>
                     <StateTable>
        622      -    <StateTableName>Main</StateTableName>
        623      -    <InitState>0</InitState>
                      <AllowableFinalStates>
                       <FinalState>18</FinalState>
                       <FinalState>40</FinalState>
        624            <FinalState>129</FinalState>
                       <FinalState>197</FinalState>
                       <FinalState>241</FinalState>
                      </AllowableFinalStates>
        625      -    <StateCount>886</StateCount>
                      <State>
(620) (621)    627 -   <StateNum>0</StateNum>
                       <OutgoingTransitions>
                        <TransitionCount>2</TransitionCount>
            (626) (628) <Transition>
                         <TransitionNum>1</TransitionNum>
                         <NextState>1</NextState>
```

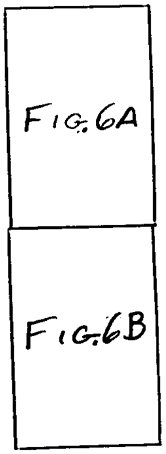


FIGURE 6

FIGURE 6A

Figure 6 (Part 1 of 2)

620 621 626 628 629

```
        <AllowableInput>
           <AllowableInputCount>1</AllowableInputCount>
           <Input>
             <DecimalCode>60</DecimalCode>
             <PrintableName>&lt;</PrintableName>
           </Input>
         </AllowableInput>
       </Transition>
       <Transition>
         <TransitionNum>2</TransitionNum>
         <NextState>2</NextState>
         <AllowableInput>
           <AllowableInputCount>4</AllowableInputCount>
           <Input>
             <DecimalCode>9</DecimalCode>
           </Input>
           <Input>
             <DecimalCode>10</DecimalCode>
           </Input>
           <Input>
             <DecimalCode>13</DecimalCode>
           </Input>
           <Input>
             <DecimalCode>32</DecimalCode>
             <PrintableName> </PrintableName>
           </Input>
         </AllowableInput>
       </Transition>
      </OutgoingTransitions>
      <PossibleIncomingTransitions>
        <IncomingTransitionCount>0</IncomingTransitionCount>
      </PossibleIncomingTransitions>
     </State>
      .
      .
      .
     <State>
            .
            .
            .
      <PossibleIncomingTransitions>
        <IncomingTransitionCount>1</IncomingTransitionCount>
        <FromStateNum>0</FromStateNum>
      </PossibleIncomingTransitions>
     </State>
      .
      .
      .
    </StateTable>
    <StateTable>
          .
          .
          .
    </StateTable>
   </StateTableDetailedInfo>
  </StateTransitionSpecification>
 </HAPC>
```

FIGURE 6B Figure 6 (Part 2 of 2)